In [1]:

```
from selenium import webdriver
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities
import time,os,traceback,sys
from selenium.common.exceptions import NoSuchElementException, ElementClickIntercept
import threading
def createDriver():
    botdriver = webdriver.Chrome(executable_path="/Users/aakash10975/chromedriver",
    botdriver.get("https://web.whatsapp.com/")
    time.sleep(5)
    return botdriver
options = webdriver.ChromeOptions()
options.add_argument("user-data-dir=/Users/aakash10975/Memory/WebWhatsAppBot")
driver=createDriver()
global statusList
statusList = {}
```

```
/Users/aakash10975/Library/Python/2.7/lib/python/site-packages/ipykern
el_launcher.py:11: DeprecationWarning: use options instead of chrome_o
ptions
  # This is added back by InteractiveShellApp.init_path()
```

Initializing

In [ ]:

```
def convert(seconds):
    seconds = seconds % (24 * 3600)
    hour = seconds // 3600
    seconds %= 3600
    minutes = seconds // 60
    seconds %= 60
    return "%d:%02d:%02d" % (hour, minutes, seconds)

TARGETLIST = ['Friend1',"Friend2"]
mynumber="+91 xxxx xxxxxx"
import datetime
def formatDt(ts=time.time()):
    st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')
    return st

def sendNotification(botdriver,message):
    print("Inside send notification function for {}".format("Person"))
    try:
        time.sleep(4)
        user = botdriver.find_element_by_xpath('//span[@title = "{}"]'.format(mynumb
        time.sleep(1)
        user.click()
        time.sleep(2)
        msg_box = botdriver.find_element_by_class_name('_1Plpp')
        msg_box.send_keys("{}".format(message))
        button=botdriver.find_element_by_class_name('_35EW6')
        button.click()
        return 1
    except NoSuchElementException:
        print("inside exception ", traceback.print_exc())
        #return sendNotification(botdriver,message)
    except RuntimeError:
        return 1
    except ElementClickInterceptedException:
        print("ElementClickInterceptedException occured.....Retrying", traceback.pri
        return sendNotification(botdriver,message)
    except TimeoutException:
        return sendNotification(botdriver,message)
statusList={}
session = []
for elem in TARGETLIST:
    statusList[elem] = ["Unavailable",time.time()]
print(statusList)
print session

#testing sendNotification function
# sendNotification(driver,"testing the function")
```

In [ ]:

In [ ]:

```
import sys

def network_call():
    sleep_secs = 5
    while True:
        for TARGET in TARGETLIST:
            try:
                msg_box = driver.find_element_by_xpath("//div[@class='_2S1VP copyabl
                msg_box.send_keys(TARGET)
                msg_box.send_keys("\n")
                time.sleep(3)
                online = str(driver.find_element_by_class_name('_3sgkv').text.encode
                # Session is active now
                if statusList[TARGET][0] != online:   #if session is inactive, mark
                    statusList[TARGET][0]="online"
                    statusList[TARGET][1]=time.time()
    #                 sendNotification(driver,"{} is online".format(TARGET))
                    sleep_secs=1
                #session is inactive now
                elif statusList[TARGET][0] == online: #if session is active, do notl
                    print("Session still Active for {}".format(TARGET))
                    print("Previous status: {}".format(statusList[TARGET][0]))
                    pass
                #unreachable code
                else:
                    print("Non reachable code printed.")
                    print("Status: {}, statusList[TARGET][0]: {}".format(status, sta
            except NoSuchElementException:
                #session in unavailable
                online = "Unavailable"
                if statusList[TARGET][0] != online:  #if session is active, mark it
                                                     # and notify over whatsapp or c
                    statusList[TARGET][0] = "Unavailable"
    #                 session.append({TARGET:
    #                                 {"start":formatDt(), "end":formatDt(statusLis
    #                                })
                    end_time = time.time()
                    session = {TARGET:
                                    {"end":formatDt(end_time), "start":formatDt(stat
                                }
                    f=open("./status.txt","a")
                    f.write(str(session))
                    f.write("\n")
                    f.close()
                    sendNotification(driver,str(session))
                    statusList[TARGET][1] = end_time
                    sleep_secs = 5
                else:   #if session is inactive, leave as is i.e, inactive
                    sys.stdout.write('\r'+str(TARGET)+str(online))
                    sys.stdout.flush()
                    pass
        time.sleep(sleep_secs)
        print("******completed an iteration*******")
        sys.stdout.write('\r'+"******completed an iteration*******")


threading.Thread(target=network_call).start()
```

In [ ]:

```
statusList
```

In [ ]:

```
session
```